# Deduction Graphs with Universal Quantification

Herman Geuvers [1]

*Institute for Computing and Information Sciences*
*Radboud University Nijmegen, The Netherlands*

Iris Loeb[2]

*Department of Mathematics and Statistics*
*University of Canterbury, New Zealand*

**Abstract**

Deduction Graphs are meant to generalise both Gentzen-Prawitz style natural deductions and Fitch style flag deductions. They have the structure of acyclic directed graphs with boxes. In [3] we have investigated the deduction graphs for minimal proposition logic. This paper studies the extension with first-order universal quantification, showing the robustness of the concept of deduction graphs.

*Keywords:* Natural deduction, universal quantification, cut-elimination.

## 1 Introduction

### *1.1 Deduction Graphs for minimal proposition logic*

In this paper we extend deduction graphs, DGs, of [3] (see also [6]), with first-order universal quantification. In [3] we have presented deduction graphs for minimal proposition logic (only implication) as a formalism for "natural deduction with sharing". This extends Gentzen-Prawitz natural deduction, in which sharing is not possible, without introducing the arbitrariness of Fitch deductions, caused by the requirement that derivations are linear. The derivations in our formalism become acyclic directed graphs with *boxes* to delimit the scope of local assumptions. The boxes are used in the $\rightarrow$-introduction rule. We repeat some definitions of [3].

**Definition 1.1** A *closed box directed graph* is a triple $\langle X, G, (\mathcal{B}_i)_{i\in I}\rangle$ where $X$ is a set of *labels*, $G$ is a directed graph where all nodes have a label in $X$ and $(\mathcal{B}_i)_{i\in I}$ is a

[1] Email: H.Geuvers@cs.ru.nl
[2] Email: I.Loeb@math.canterbury.ac.nz

*This paper is electronically published in*
*Electronic Notes in Theoretical Computer Science*
*URL:* www.elsevier.nl/locate/entcs

collection of sets of nodes of $G$, the *boxes*. Each box $\mathcal{B}_i$ corresponds to a node, the *box node* of $\mathcal{B}_i$. Moreover, the boxes $(\mathcal{B}_i)_{i\in I}$ should satisfy the following properties.

(i) (Non-overlap) Two boxes are disjoint or one is contained in the other: $\forall i,j \in I(\mathcal{B}_i \cap \mathcal{B}_j = \emptyset \vee \mathcal{B}_i \subset \mathcal{B}_j \vee \mathcal{B}_j \subset \mathcal{B}_i)$,

(ii) (box node edge) There is only one outgoing edge from a box node and that points into the box itself (i.e. to a node in the box),

(iii) (No edges into a box) Apart from the edge from the box node, there are no edges pointing into a box.

**Definition 1.2** Let $\langle G, (\mathcal{B}_i)_{i\in I}\rangle$, be a closed box directed graph and let $n_0$ and $n_1$ be nodes in this graph.

- Node $n_1$ is *in scope of* $n_0$ if $n_0$ is in all boxes that $n_1$ is in. In a formula: $\forall i \in I(n_1 \in \mathcal{B}_i \Rightarrow n_0 \in \mathcal{B}_i)$. (So the nodes in scope of $n_0$ are the nodes that are in 'wider' boxes.)
- The nodes $n_0$ and $n_1$ are *at the same depth*, when $n_0$ is in scope of $n_1$, and $n_1$ is in scope of $n_0$. Node $n_0$ is *at a greater depth* than $n_1$, when $n_1$ is in scope of $n_0$, but $n_0$ is not in scope of $n_1$.
- Node $n_1$ is a *top-level node* if $n_1$ is not contained in any box.
- The *free nodes* are the top-level nodes that have no outgoing edges.

The importance of the next definition will become clear in Lemma 2.6.

**Definition 1.3** Let $G$ be a closed box directed graph. A *box-topological ordering* of $G$ is a linear ordering $<$ of the nodes of $G$, such that for all nodes $n_0, n_1$ of $G$:

(i) If $n_0 \longrightarrow\!\triangleright\, n_1$, then $n_1 < n_0$.

(ii) If $n_0$ is the box node of a box containing $n_1$, then $n_1 < n_0$.

Deduction graphs are a special kind of closed box directed graphs (Definition 1.1): The labels of the nodes are formulas of minimal proposition logic, and there are rules to ensure that the graph represents a logical derivation. (The precise definition can be found in [3]; in the present paper it appears as part of Definition 2.3.) Figure 1 shows an example.

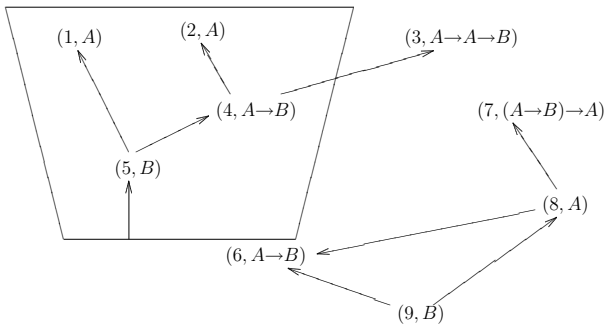



Fig. 1. Deduction graph in the implicational fragment

The arrow represents (inverse) derivability, so e.g. node $(9, B)$ is derived from nodes $(6, A{\to}B)$ and $(8, A)$: It is an $\to$-elimination. Similarly node $(6, A{\to}B)$ is derived from $(5, B)$ while discharging the nodes without outgoing edges (assumptions) $(1, A)$ and $(2, A)$: It is an $\to$-introduction. Note that node $(6, A{\to}B)$ is *shared*: It has two incoming edges.

Deduction graphs can contain *cuts*: An $\to$-introduction, immediately followed by an $\to$-elimination (possibly separated by the repetition of formulas). In principle, we can eliminate a cut in the way that is indicated schematically in Fig.2.

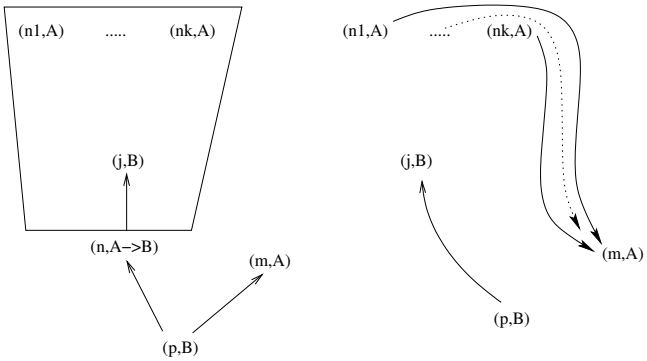



Fig. 2. Cut-elimination in deduction graphs.

However, the actual situation is harder than the one sketched in Figure 2:

(i) The $\to$-introduction and the $\to$-elimination may be separated by repeats.

(ii) The involved box may have multiple incoming edges. In this case, if we would transform the deduction graph as indicated in the figure, the resulting structure would not be a deduction graph again, as these edges would be "dangling".

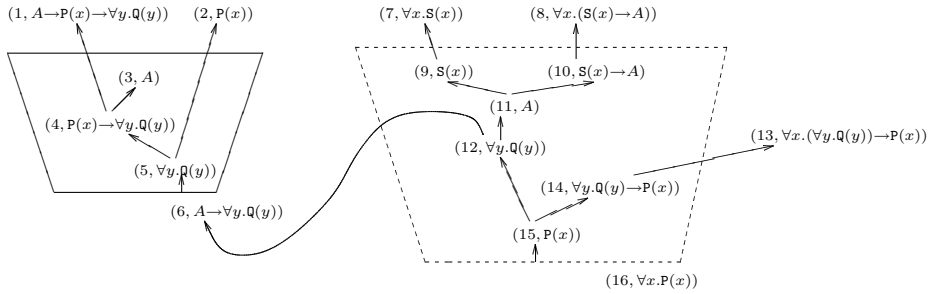
(iii) The minor premiss $(m, A)$ of the elimination might be at a greater depth than the major premiss $(n, A{\to}B)$. In this case, if we would transform the deduction graph as indicated in the figure, the resulting graph may not be a closed box directed graph, as edges from a node $((n_1, A), \ldots, (n_k, A)$ in the figure) might go to a node at a greater depth $((m, A))$.

If these problems do not occur, we call the cut *safe*. The transformations *repeat-elimination*, *unsharing*, and *incorporation* on deduction graphs are used to make a cut safe if any of the above problems occurs. (See [3]). The series of transformations involved in making a cut safe and eliminating the safe cut, is called *the process of cut-elimination*. In [3] we have shown by going to a $\lambda$-calculus with tupling, that the process of cut-elimination is *strongly normalising*, i.e. we can transform any deduction graph into a cut-free deduction graph in a finite numbers of steps using these transformations.

### *1.2 Adding Universal Quantification*

Originally, boxes were meant to border the scope of a local assumption, but now we also use boxes to border the scope of a quantifier: When we do a $\forall$-introduction, we create a box with box node $\forall x.\varphi$. To carry this extension through we have to

consider how to deal with the side condition on the ∀-introduction rule, which is stated in Gentzen-Prawitz style natural deduction as follows: "the *eigenvariable* does not occur free in any of the non-discharged assumptions". (The eigenvariable is the quantified variable $x$ in the introduction of $\forall x.\varphi$.) In $\mathtt{DG_{US}}$ we want to represent this by a more "local" side condition. A first idea would be to require that *there is no edge pointing out of the box to a formula in which the eigenvariable occurs free*, like usually done in Fitch deductions. (So when we introduce $\forall x.\varphi$, the box we create should not have edges pointing out to a node $\psi$ with $x \in \mathrm{FV}(\psi)$.) However, this would cause severe problems in the cut-elimination procedure, as the following graph shows. The ∀-box has been depicted with a dashed line.




There is a non-safe →-cut in node 12: Nodes 6 and 12 are not at the same depth. So we first have to do an incorporation step, moving the box with box node 6 into the box with box node 16.

The eigenvariable of the ∀-box is $x$. If we would do an incorporation directly, there would be arrows from inside the ∀-box to the nodes 1 and 2 outside the box, in which $x$ occurs free. This is forbidden. We therefore first have to do a renaming of the eigenvariable, as shown in Fig. 3.

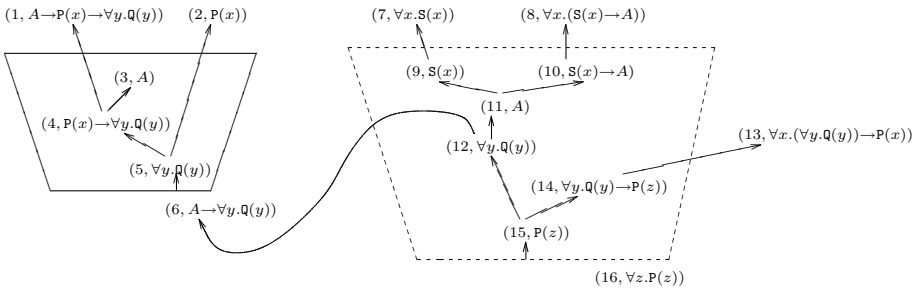



Fig. 3. Renaming of eigenvariable.

This renaming is not so trivial because it not only involves nodes inside the box but also the $x$ in node 16. But when we rename $x$ in node 16, we also have to rename it in nodes that refer to 16, and propagate that through the graph. This could thus involve any node of $G$, eventually even nodes 1 and 2. Renaming is hence not just complicated, but it might a priori not even solve the problem.

As this looks like Gentzen-Prawitz style natural deduction, why doesn't the necessity to rename variables occur there? There is no sharing in the example

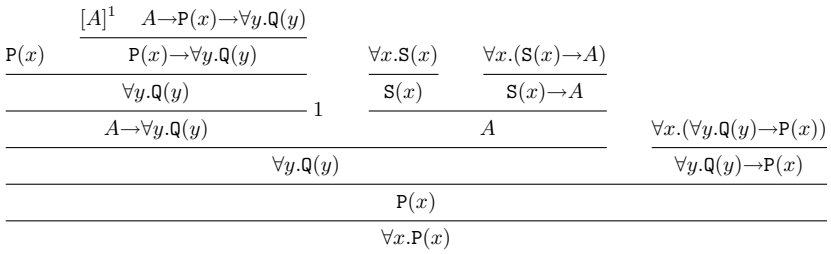



Fig. 4. Gentzen-Prawitz style natural deduction (incorrect).

graph, so we can present the deduction faithfully as a tree as done in Fig.4.

However, this is not a correct Gentzen-Prawitz style natural deduction, as the variable $x$ occurs free in some non-discharged assumptions when it gets bound. Apparently the $\forall$-introduction rule in Gentzen-Prawitz style natural deduction is – due to its "global" nature – strict enough to prevent the need for renaming variables during $\forall$-cut-elimination.

Our solution is to use two sets of variables: free variables, $\mathsf{Var}$ and bound variables $\mathsf{BVar}$ and to rename the free variable with a fresh bound variable when doing the $\forall$-introduction. Furthermore, we require that the eigenvariable is unique for that box (i.e. it does not occur anywhere outside the box). A further discussion of the choice of syntax can be found in Section 3.1.

In Section 2 we give the definition of deduction graphs with universal quantification, called $\mathtt{DG_U s}$, starting from definitions for terms and formulas of first-order predicate logic. The process of cut-elimination is discussed in Section 3, followed by strong normalisation in Section 4. Finally, Section 5 compares $\mathtt{DG_U s}$ with developments in proof nets.

## 2 Definition

Different from the language of first-order predicate logic for Gentzen-Prawitz style natural deduction [1,7], we define the language $\mathsf{Pred}$ of first-order predicate logic with universal quantification and equality for deduction graphs to have two kinds of variables. The first kind, $\mathsf{Var}$ denoted by $u, v, w, \ldots$, are meant to be used as *free* variables. The second kind, $\mathsf{BVar}$, denoted by $x, y, z, \ldots$, will only be used *bound*. The same idea is often used for the language of first-order predicate logic for Fitch style flag deductions [8].

We now define the terms, basic formulas, formulas and axioms of $\mathsf{Pred}$.

**Definition 2.1** (i) The set $\mathsf{Term}$ of *terms* of $\mathsf{Pred}$, $\mathsf{Term}$ is defined as follows.

$$\mathsf{Term} ::= \mathsf{Var} \mid \mathtt{F}(\mathsf{Term}, \ldots, \mathsf{Term})$$

where $\mathtt{F}$ is a function symbol with a fixed arity and the length of the sequence of terms following it should be equal to the arity of $\mathtt{F}$.

(ii) The set of *formulas* of $\mathsf{Pred}$, $\mathsf{Form}$, is defined as follows.

$$\mathsf{Form} ::= \mathsf{R}(\mathsf{Term}, \ldots, \mathsf{Term}) \mid \mathsf{Term} = \mathsf{Term} \mid \mathsf{Form} \to \mathsf{Form} \mid \forall x.\mathsf{Form}[x/u]$$

where $\mathsf{R}$ is a relation symbol with a fixed arity and the length of the sequence of terms following it should be equal to the arity of $\mathsf{R}$; $x$ ranges over $\mathsf{BVar}$ and $u$ over $\mathsf{Var}$ and we require that $x$ is *fresh*, i.e. it does not yet occur as a bound variable in the formula.

**Remark 2.2** When we introduce a binder we also replace a free variable by a bound one. This replacement is denoted by the substitution $[x/u]$: if $\varphi \in \mathsf{Form}$ and $x$ is not a bound variable in $\varphi$, then $\forall x.\varphi[x/u] \in \mathsf{Form}$. Because $x$ is required to be fresh, the substitution $[x/u]$ does not involve a renaming of bound variables.

We note that, due to our distinction between $\mathsf{BVar}$ and $\mathsf{Var}$ and the requirement that bound variables occur uniquely in a formula, the substitutions $\psi[t/x]$ (replacing $x$ by $t$ to be able to go from $\forall x.\psi$ to $\psi[t/x]$), $\varphi[x/u][t/x]$ (used in the definition of cut-elimination later) and $\varphi[t/u]$ are defined unambiguously.

We adopt the following convention for the brackets in formula: $\to$ is right-associative; $\forall$ binds stronger than $\to$; outer brackets are not written, nor are any other brackets that do not contribute to our understanding of the formula.

So, for example, $\forall x.\varphi \to \psi \to \xi \equiv ((\forall x.\varphi) \to (\psi \to \xi))$. Note, however, that $\forall x.\mathtt{P}(x) \to \mathtt{Q}(x)$ can formally only be understood as $(\forall x.(\mathtt{P}(x) \to \mathtt{Q}(x)))$, because $((\forall x.\mathtt{P}(x)) \to \mathtt{Q}(x))$ is not a formula. In these cases we will write the inner brackets under the quantifier explicitly anyway, for the convention would otherwise lead us to misinterpret the formula.

Because $\mathsf{Pred}$ deviates from the language of first-order predicate logic for Gentzen-Prawitz natural deduction, this also means that the $\forall$-introduction for deduction graphs cannot be similar to the $\forall$-introduction in the Gentzen-Prawitz formalism.

The $\forall$-introduction in Gentzen-Prawitz style natural deduction is as follows:

$$\dfrac{\dfrac{D}{\varphi}}{\forall x.\varphi}$$

Where $x$ may not be free in the non-discharged assumptions of $D$. This means that $x$ might be free in $\varphi$, although it is bound in $\forall x.\varphi$.

In deduction graphs we will introduce a fresh (bound) variable in the $\forall$-introduction step. The advantage is then, that a graph may still be correct, when we rename only free variables. We will use this later, in the process of cut-elimination.

**Definition 2.3** The collection of *deduction graphs for first-order universal quantification*, $\mathtt{DG_U}$ is the set of closed box directed graphs over $\mathbf{N} \times \mathsf{Pred}$ inductively defined as follows:

**Axiom** A single node $(n, A)$ is a deduction graph,

**→-E** If $G$ is a deduction graph containing two nodes $(n, A{\to}B)$ and $(m, A)$ at the top level, then the graph $G' := G$ with
· a new node $(p, B)$ at the top level
· an edge $(p, B) \longrightarrow\!\triangleright (n, A{\to}B)$,
· an edge $(p, B) \longrightarrow\!\triangleright (m, A)$,
is a deduction graph.

**→-I** If $G$ is a deduction graph containing a node $(j, B)$ with no ingoing edge and a finite set of free nodes with label $A$, $(n_1, A), \ldots, (n_k, A)$, all at the top level, then the graph $G' := G$ with
· a box $\mathcal{B}$ with box node $(n, A{\to}B)$, containing the nodes $(j, B)$ and $(n_1, A)$, $\ldots, (n_k, A)$ and no other nodes that were free in $G$,
· an edge from the box node $(n, A{\to}B)$ to $(j, B)$
is a deduction graph under the proviso that it is a closed box directed graph.

**Repeat** If $G$ is a deduction graph containing a node $(n, A)$ at the top level, the graph $G' := G$ with
· a new node $(m, A)$ at the top level,
· an edge $(m, A) \longrightarrow\!\triangleright (n, A)$
is a deduction graph.

**∀-I** If $G$ is a $\mathtt{DG_U}$ containing a node $(j, \varphi)$ with no ingoing edge at top-level for some formula $\varphi$ of $\mathsf{Pred}$, then the graph $G' := G$ with
· a box $\mathcal{B}$ with box node $(n, \forall x.\varphi[x/u])$ , not containing any nodes without outgoing edges, where we call $u$ the *eigenvariable* of $\mathcal{B}$ if $u$ occurs in $\varphi$,
· an edge from the box node $(n, \forall x.\varphi[x/u])$ to $(j, \varphi)$
is a $\mathtt{DG_U}$ under the proviso that: $G'$ is a well-formed closed box directed graph and $u$ does not occur in the label of any node that is not in $\mathcal{B}$.

**∀-E** If $G$ is a $\mathtt{DG_U}$ with a node $(n, \forall x.\varphi)$ at top-level for some formula $\varphi$ of $\mathsf{Pred}$, then the graph $G' := G$ with
· a node $(p, \varphi[t/x])$ where none of the variables of $t$ is the eigenvariable of any box of $G$,
· an edge from $(p, \varphi[t/x])$ to $(n, \forall x.\varphi)$
is a $\mathtt{DG_U}$.

**Join$_\mathtt{U}$** If $G$ and $G'$ are two $\mathtt{DG_U}$s then $G'' = G \cup G'$ is a $\mathtt{DG_U}$ under proviso that the eigenvariables of $G$ and the eigenvariables of $G'$ are disjoint.

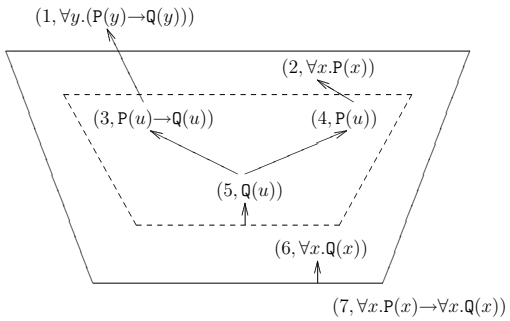



Fig. 5. An example of a $\mathtt{DG_U}$.

So the rules for $\mathtt{DG_U}$ are the ones for $\mathtt{DG}$ with the $\forall$-I and $\forall$-E rules added and the Join rule slightly modified.

**Example 2.4** Let $\mathtt{P}$ and $\mathtt{Q}$ be unary predicate symbols of $\mathsf{Pred}$. Figure 5 shows an example of an $\mathtt{DG_U}$.

**Lemma 2.5** *Let $G$ be a $\mathtt{DG_U}$. Then for every variable $u$:*

(i) *$u$ occurs as eigenvariable of a box of $G$ at most once;*

(ii) *If $u$ is an eigenvariable of a box $\mathcal{B}$ of $G$, it does not occur in a label of a node outside $\mathcal{B}$.*

We formulate a criterion to check relatively easily whether a given closed box directed graph is a $\mathtt{DG_U}$ (Lemma 2.6). As an important notion for $\mathtt{DG_U}$s is the eigenvariable of a box, we need a similar notion for general closed box directed graphs. So, we call a variable $u$ a *box-variable of $\mathcal{B}$*, if $u$ does not occur in the label of the box node of $\mathcal{B}$, but it does occur in the label of the node that the box node points to. Remark that for $\mathtt{DG_U}$s the notion of eigenvariable and the notion of box-variable coincide. We also recall from [3] the notion of a *box-topological ordering*: $>$ is a box-topological ordering of $G$ if it is a linear ordering of the nodes of $G$, such that $n \longrightarrow\!\!\triangleright\, m \Rightarrow n > m$ and if $\mathcal{B}$ has box node $n$ and $m \in \mathcal{B}$, then $n > m$.

**Lemma 2.6** *A closed box directed graph $G$ is a $\mathtt{DG_U}$ if and only if the following hold*

(i) *If $u$ is a box-variable of a box $\mathcal{B}$ of $G$, it does not occur in a label of a node outside $\mathcal{B}$.*

(ii) *$G$ is finite.*

(iii) *There is a box-topological ordering $>$ of $G$.*

(iv) *Every node of $G$ is of one of the following six types:*

- A *$n$ has no outgoing edges.*
- $\to$-E *$n$ has label $B$ and has exactly two outgoing edges: one to a node $(m, A{\to}B)$ and one to a node $(p, A)$, both within the scope of $n$.*
- $\to$-I *$n$ is a box node of a box $\mathcal{B}$ with label $A{\to}B$ and has exactly one outgoing edge, which is to a node $(j, B)$ inside the box $\mathcal{B}$ (and not in any deeper boxes) with no other ingoing edges. All nodes inside the box without outgoing edges have label $A$.*
- R *$n$ has label $A$ and has exactly one outgoing edge, which is to a node $(m, A)$ that is within the scope of $n$.*
- $\forall$-E *$n$ has label $\varphi[t/x]$ for some formula $\varphi$, some term $t$ and some variable $x$, and $n$ has exactly one outgoing edge: to a node $(m, \forall x.\varphi)$ within the scope of $n$.*
- $\forall$-I *$n$ is a box node of a box $\mathcal{B}$ with label $\forall x.\varphi$ for some variable $x$ and some formula $\varphi$, and has exactly one outgoing edge, which is to a node $(j, \varphi[u/x])$ inside the box $\mathcal{B}$ (and not in any deeper boxes) with no other ingoing edges. There are no nodes without outgoing edges in $\mathcal{B}$.*

## 3 Cut-elimination

By introducing the universal quantification, we have also introduced a new cut, the $\forall$-*cut* (as opposed to the $\to$-cut): A $\forall$-introduction immediately followed by a

$\forall$-elimination. Lemma 3.3 describes the elimination of a safe $\forall$-cut.

Just as not all $\to$-cuts are safe, neither are all $\forall$-cuts, so it might be necessary to apply some transformations to make them safe. These transformations are the same ones as for $\to$-cuts: repeat-elimination, unsharing, and incorporation. The only difference with the transformations on DGs is, that unsharing has become a little more involved, because of the eigenvariable requirement.

**Definition 3.1** A $\forall$-*cut* in a $\mathtt{DG_U}$ $G$ is a subgraph of $G$ consisting of:

- a box node $(n, \forall x.\varphi)$,
- a node $(p, \varphi[t/x])$,
- a sequence of R-nodes $(s_0, \forall x.\varphi), \ldots, (s_i, \forall x.\varphi)$,
- edges $(p, \varphi[t/x]) \longrightarrow\!\triangleright (s_i, \forall x.\varphi) \longrightarrow\!\triangleright \ldots \longrightarrow\!\triangleright (s_0, \forall x.\varphi) \longrightarrow\!\triangleright (n, \forall x.\varphi)$.

We call the node $(n, \forall x.\varphi)$ the *major premiss* and we call the node $(p, \varphi[t/x])$ the *conclusion* of the $\forall$-cut.

Similarly, in a $\to$-cut, we call $(n, A \to B)$ the major premiss and the node $(p, B)$ the conclusion.

**Definition 3.2** Let $\mathcal{B}$ be the box associated to box node $n$. A $(\forall/\to)$-cut in a $\mathtt{DG_U}$ $G$ is *safe* if the following requirements hold:

- there is an edge from the conclusion to the major premiss and that is the only edge to the major premiss;
- the major premiss and the conclusion are at the same depth (relative to the box structure);

**Definition 3.3** The process of *eliminating a safe* $\forall$-*cut* is the following operation on $\mathtt{DG_U}$s (see Figure 6):

- change the labels $\psi$ of the nodes in the box of $n$, to $\psi[t/u]$ [3];
- remove the box and box node $(n, \forall x.\varphi[x/u])$;
- add an edge from $(p, \varphi[t/u])$ to $(j, \varphi[t/u])$ (the node that $n$ pointed to).

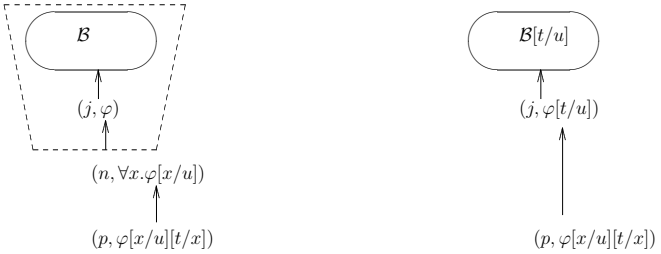



Fig. 6. Schematic presentation of a safe $\forall$-cut elimination.

**Lemma 3.4** *If $G$ is a $\mathtt{DG_U}$ with safe $\forall$-cut $c$ and $G'$ is obtained from $G$ by eliminating $c$, then $G'$ is also a $\mathtt{DG_U}$.*

[3] Substituting $t$ for $u$ in $\psi$ does not involve renaming, because $t$ only contains free variables. See also Remark 2.2.

**Proof.** By Lemma 2.6. □

We can generalise repeat-elimination, unsharing, and incorporation without much ado. Because after unsharing we still want every eigenvariable to occur just once, this step now includes the renaming of eigenvariables of copied boxes.

**Definition 3.5** Let $G$ be a $\mathtt{DG_U}$ with a cut with major premiss $(n, \varphi)$ and conclusion $(p, \psi)$. Suppose $G$ contains a node $(n_0, \varphi)$, an $\mathsf{R}$-node $(n_1, \varphi)$ and edges $n_1 \longrightarrow\!\triangleright\, n_0$ and $p \longrightarrow\!\triangleright\, n_1$. The *repeat-elimination* at $n_0, n_1, p$ is obtained by:

- When an edge points to $n_1$, redirect it to $n_0$;
- Remove $n_1$.

**Lemma 3.6** *For $G$ a $\mathtt{DG_U}$ with a cut with major premiss $(n, \varphi)$ and conclusion $(p, \psi)$. Suppose $G$ contains a node $(n_0, \varphi)$, an $\mathsf{R}$-node $(n_1, \varphi)$ and edges $n_1 \longrightarrow\!\triangleright\, n_0$ and $p \longrightarrow\!\triangleright\, n_1$, the repeat-elimination of at $n_0, n_1, p$ is also a $\mathtt{DG_U}$.*

**Proof.** By Lemma 2.6. □

**Definition 3.7** Let $G$ be a $\mathtt{DG_U}$ with a $\forall$-box $\mathcal{B}$ with eigenvariable $u$. Let $v$ be a fresh variable. Then the *renaming of $u$ by $v$* is the graph $G$ in which the labels $\psi$ of the nodes of $\mathcal{B}$ have been replaced by $\psi[v/u]$.

**Lemma 3.8** *Let $G$ be a $\mathtt{DG_U}$ with a $\forall$-box $\mathcal{B}$ with eigenvariable $u$. Let $v$ be a fresh variable. Then the renaming of $u$ by $v$ is a $\mathtt{DG_U}$.*

**Proof.** By Lemma 2.6. □

**Definition 3.9** Let $G$ be a $\mathtt{DG_U}$ with a cut $c$ with major premiss $n$. Suppose $n$ is a box node of a box $\mathcal{B}$ and has $k \geq 2$ ingoing edges, from $p_1, \ldots, p_k$. Then the *unsharing of $G$ at nodes $n, p_1, \ldots p_n$* is obtained by:

- making a box $\mathcal{B}'$ that contains a copy of all nodes and edges of $\mathcal{B}$,
- copy all outgoing edges of $\mathcal{B}$ to $\mathcal{B}'$ (thus if we had $q \longrightarrow\!\triangleright\, m$ with $q \in \mathcal{B}$, $q' \in \mathcal{B}'$ and $m \notin \mathcal{B}$, then we add $q' \longrightarrow\!\triangleright\, m$, where $q'$ is the copy of $q \in \mathcal{B}'$,
- letting $p_2, \ldots, p_k$ point to $n'$ (the box node of $\mathcal{B}'$) instead of $n$;
- renaming the eigenvariable of $\mathcal{B}'$ and of all boxes contained in $\mathcal{B}'$.

**Lemma 3.10** *Let $G$ be a $\mathtt{DG_U}$ with a cut $c$ with major premiss $n$. Suppose $n$ is a box node of a box $\mathcal{B}$ and has $k \geq 2$ ingoing edges, from $p_1, \ldots, p_k$. Then the unsharing of $G$ at nodes $n, p_1, \ldots p_n$ is a $\mathtt{DG_U}$.*

**Proof.** By Lemma 2.6. □

**Definition 3.11**
We have a *depth-conflict* in the $\mathtt{DG_U}$ $G$, if $G$ contains a cut with major premiss $n$ and conclusion $p$ at a greater depth, such that there is an arrow from $p$ to $n$ and that is the only arrow to $n$. In that case the *incorporation* of $G$ at $n, p$ is obtained by moving $\mathcal{B}_n$, i.e. the box of $n$, into the box at the lowest depth that includes $p$ but excludes $n$.

**Lemma 3.12** *Suppose $G$ is a $\mathtt{DG_U}$ with a depth conflict. Then the incorporation at the major premiss and the conclusion is a $\mathtt{DG_U}$.*

**Proof.** By case analysis on the incorporating box. Then by Lemma 2.6. □

**Definition 3.13** Given a $\mathtt{DG}_{\mathtt{U}}$ $G$ with a cut $c$, the *process of $\to/\forall$-cut elimination* is the following;

(i) (Repeat elimination) As long as there is no edge from the conclusion to the major premiss, perform the appropriate repeat-elimination as defined in Definition 3.5;

(ii) (Unsharing) If there is an edge from the conclusion to the major premiss, but this is not the only edge to the major premiss, perform an appropriate unsharing step, as defined in Definition 3.9;

(iii) (Incorporation) As long as the conclusion is at a greater depth than the major premiss, perform the appropriate incorporation step, as defined in Definition 3.11.

(iv) (Eliminating a safe cut) If $c$ is safe, perform either the safe $\to$-cut-elimination step, or the safe $\forall$-cut-elimination step, as defined in Definition 3.3.

### *3.1 Discussion*

We have made some choices in the definition of $\mathtt{DG}_{\mathtt{U}}$s that facilitate the process of cut-elimination. Except for the choice of the language, which has already been discussed in the Introduction, these are:

(i) We deviate from the side-condition for the $\forall$-introduction rule as normally used in Fitch-style flag deduction, as discussed in the Introduction. (For deduction graphs, this reads as: "there is no edge pointing out of the box to a formula in which the eigenvariable occurs free.)

(ii) We require the uniqueness of the eigenvariables.

Suppose we would adopt the Fitch-style side-condition for the $\forall$-introduction rule, then this results in having to do an additional renaming in the incorporation step in some cases.

If we would abandon the requirement of unique eigenvariables and adopt the Fitch-style side-condition, this would move renaming from the unsharing step to the incorporation step.

## 4 Strong Normalisation

To obtain strong normalisation for cut-elimination on $\mathtt{DG}_{\mathtt{U}}$s, we extend the $\lambda$-calculus with tupling as defined in [3], and prove strong normalisation for it. Then a reduction preserving translation from $\mathtt{DG}_{\mathtt{U}}$s to this calculus is defined.

The strong normalisation result we thus get is relatively weak: It is assumed that first one cut is made safe and is eliminated, before handling another cut.

For Gentzen-Prawitz natural deduction, strong normalisation for cut-elimination can be proven by (1) defining a $\to$-cut preserving translation to the $\to$-fragment and (2) showing that an infinite $\forall$-cut reduction is impossible. That might also work for the $\mathtt{DG}_{\mathtt{U}}$ case, but (2) is now problematic, because a $\forall$-cut contraction may involve unsharing and then other $\forall$-cuts may be copied. We therefore opt for a direct proof

of strong normalisation for cut-elimination for $\mathtt{DG_US}$.

**Definition 4.1** The typed *expressions* $\mathsf{T}_{\langle\rangle}$ and *types* of the $\lambda{\to}\langle\rangle$*-calculus* for first order predicate logic with universal quantification are defined as follows.

(i) For $\varphi \in \mathsf{Form}$, all variables $x^\varphi$ are of type $\varphi$.

(ii) If $T$ is of type $\varphi{\to}\psi$ and $S$ is of type $\varphi$, $(TS)$ is of type $\psi$.

(iii) If $T$ is of type $\varphi$, then $\lambda x^\psi.T$ is of type $\psi{\to}\varphi$.

(iv) If $T$ is of type $\forall x.\varphi$ and $t$ is a term, then $(Tt)$ is of type $\varphi[t/x]$.

(v) If $T$ is of type $\varphi$, then $\lambda y.T[u := y]$ is of type $\forall y.\varphi[y/u]$.

(vi) If $T_1, \ldots, T_n$ are of types $\varphi_1, \ldots, \varphi_n$ respectively, then $\langle T_1, \ldots, T_n\rangle$ is an expression of type $\varphi_1$.

**Definition 4.2** The reduction rules for the expressions are as follows:

$$\begin{aligned}
(\lambda x^\sigma.M)N &\longrightarrow\!\!\!\!\!\rightarrow_{\bar\beta} \langle M, N\rangle \text{ if } x \notin \mathrm{FV}(M)\\
(\lambda x^\sigma.M)N &\longrightarrow\!\!\!\!\!\rightarrow_{\bar\beta} M[x := N] \text{ if } x \in \mathrm{FV}(M)\\
(\lambda y.M)t &\longrightarrow\!\!\!\!\!\rightarrow_{\bar\beta} M[y := t]\\
\langle M, P_1, \ldots, P_k\rangle N &\longrightarrow\!\!\!\!\!\rightarrow_{\bar\beta} \langle MN, P_1, \ldots, P_k\rangle\\
\langle M, P_1, \ldots, P_k\rangle t &\longrightarrow\!\!\!\!\!\rightarrow_{\bar\beta} \langle Mt, P_1, \ldots, P_k\rangle\\
N\langle M, P_1, \ldots, P_k\rangle &\longrightarrow\!\!\!\!\!\rightarrow_{\bar\beta} \langle NM, P_1, \ldots P_k\rangle\\
\langle \ldots, \langle M, P_1, \ldots, P_k\rangle, \ldots\rangle &\longrightarrow\!\!\!\!\!\rightarrow_{\bar\beta} \langle \ldots, M, P_1, \ldots, P_k, \ldots\rangle
\end{aligned}$$

As can be observed from the typing and the reduction rules, the $N_1, \ldots, N_k$ in $\langle M, N_1, \ldots, N_k\rangle$ act as a kind of 'garbage'. The order of the typed expressions in $N_1, \ldots, N_k$ is irrelevant and we therefore consider terms *modulo* permutation of these vectors, which we will write as $\equiv_p$.

**Definition 4.3** Given a deduction graph $G$ and a node $n$ in $G$, we define the $\lambda$-term $\langle\!|G, n|\!\rangle$ as follows (by induction on the number of nodes of $G$).

$\mathsf{A}$ If $(n, A)$ has no outgoing edges, $\langle\!|G, n|\!\rangle := x_n^A$,

$\to\mathsf{E}$ If $(n, B) \longrightarrow\!\!\!\!\!\triangleright\, (m, A{\to}B)$, and $(n, B) \longrightarrow\!\!\!\!\!\triangleright\, (p, A)$, define $\langle\!|G, n|\!\rangle := \langle\!|G, m|\!\rangle\, \langle\!|G, p|\!\rangle$.

$\mathsf{R}$ If $(n, A) \longrightarrow\!\!\!\!\!\triangleright\, (m, A)$, define $\langle\!|G, n|\!\rangle := \langle\!|G, m|\!\rangle$

$\to\mathsf{I}$ If $(n, A{\to}B)$ is a box node with $(n, A{\to}B) \longrightarrow\!\!\!\!\!\triangleright\, (j, B)$, the free nodes of the box are $n_1, \ldots, n_k$ and the nodes without incoming edges inside the box are $m_1, \ldots, m_p$, then

$$\langle\!|G, n|\!\rangle := \lambda x^A.\langle\langle\!|G, j|\!\rangle, \langle\!|G, m_1|\!\rangle, \ldots, \langle\!|G, m_p|\!\rangle\rangle[x_{n_1} := x, \ldots, x_{n_k} := x].$$

$\forall\mathsf{E}$ If $(n, \varphi[t/y]) \longrightarrow\!\!\!\!\!\triangleright\, (p, \forall y.\varphi)$, define $\langle\!|G, n|\!\rangle := \langle\!|G, p|\!\rangle t$.

$\forall\mathsf{I}$ If $(n, \forall y.\varphi[y/u]) {\to\!\!\!\!\!\triangleright} (j, \varphi)$ and the nodes without incoming edges are $m_1, \ldots, m_p$, then

$$\langle\!|G, n|\!\rangle := \lambda y.\langle\langle\!|G, j|\!\rangle, \langle\!|G, m_1|\!\rangle, \ldots, \langle\!|G, m_p|\!\rangle\rangle[u := y]$$

The interpretation of the deduction graph $G$, $\langle\!|G|\!\rangle$, is defined as $\langle\langle\!|G, r_1|\!\rangle, \ldots, \langle\!|G, r_l|\!\rangle\rangle$, where $r_1, \ldots, r_l$ are the top-level nodes without incoming edges in the deduction graph $G$.

**Definition 4.4** A $\lambda{\to}\langle\rangle$-*context* is given by the following abstract syntax $K[-]$:

$$K[-] := [-] \mid \mathsf{T}_{\langle\rangle} K[-] \mid K[-]\mathsf{T}_{\langle\rangle}$$

So a $\lambda{\to}\langle\rangle$-context is a $\lambda{\to}\langle\rangle$-term consisting only of applications (no abstractions) with one open place. The following is by induction on $K[-]$.

**Lemma 4.5** *For all $\lambda{\to}\langle\rangle$ contexts $K[-]$ and $\lambda{\to}\langle\rangle$-terms $M, N_1, \ldots, N_k$*

$$K[\langle M, N_1, \ldots, N_k\rangle] \twoheadrightarrow_{\bar\beta} \langle K[M], N_1, \ldots, N_k\rangle.$$

**Lemma 4.6 ($\forall$ Cut-elimination is $\bar\beta$-reduction in $\lambda{\to}\langle\rangle$)**
*If $G'$ is obtained from $G$ by a $\forall$-cut-elimination, then $(\!|G|\!) \twoheadrightarrow^{+}_{\bar\beta} (\!|G'|\!)$.*

**Proof.** By induction on the structure of $G$. Similar to the DG case. □

**Theorem 4.7** *The process of cut-elimination is terminating for $\mathtt{DG}_\mathtt{U}$s.*

**Proof.** Analogous to the DG case. □

## 5 Connection with Proof Nets

In [2] we have seen a correspondence between a variant of DGs and proof nets of MELL. We remarked that there are some superficial similarities between the two: both have boxes and both enable sharing (contraction). Using this, we were able to define a translation from these deduction graphs to proof nets that preserves reduction.

In the way they handle quantification proof nets also seem fairly close to deduction graphs. In the early days [4] boxes were used to delimit the scope of a quantification. Later (see for example [9]), this was put aside and replaced by global correctness criteria. It seems plausible that in deduction graphs too we could omit boxes for this use. We have not done this as deduction graphs serve another purpose than proof nets, and leaving out the $\forall$-boxes would make the deduction graphs less perspicuous. This discrepancy in the handling of quantification does not seem to jeopardise the aim to extend the translation given in [2]: Because $(\forall x.\varphi)^* = !(\forall x.\varphi^*)$ (where $()^*$ is Girard's translation), it is the *exponential* box that should act like the $\forall$-boxes in deduction graphs anyway.

The main difficulty in both proof nets and deduction graphs is that during cut-elimination it is in some cases necessary to do a renaming. In anticipation to this, we have changed the $\forall$-introduction rule for deduction graphs and we have used two kinds of variables: one kind for bound uses, and one for free uses (see also [8]).

In [5], Girard discusses proof nets of MLL with quantifiers. Note that, as these proof nets do not include the exponential rules, this results in a simpler system. His approach is similar to ours. He replaces some free variables by constants, which reminds of our solution with two kinds of variables. He also insists on uniqueness of the eigenvariable. About renaming he says:

In practice, it would be crazy to rename bound variables (...).

Luckily, as there is no copying going on in the cut-elimination of MLL, renaming is nowhere necessary.

This changes when we shift our attention to MELL proof nets with quantification. The most complete study of this can be found in [9], and although it handles only second order quantification explicitly, it is generally assumed [4,9], that first order quantifiers do not provide additional difficulties. Instead of discriminating between different kinds of variables, in [9] an equivalence relation is defined on the formulas, making two formulas the same when one can be obtained from the other by renaming bound variables. Deviating from [9], this line might be pursued as follows [4]:

(i) Define formulas;

(ii) Define proof-structures;

(iii) Define the equivalence relation on formulas;

(iv) Extend the equivalence relation on proof-structures.

Once this has been done, it needs to be shown that after cut-elimination on a proof-net, one gets a proof-structure that is equivalent to a proof net.

This plan has two difficulties, the first being the exact definition of the equivalence relation on proof structures. Just saying that two proof structures are equivalent, when they have the same structure and when formulas at the same places are equivalent, would not suffice. In addition, it should also consider renaming of *free* variables in formulas that will get bound somewhere else in the structure.

Secondly, it could be rather complicated to find an equivalent proof-net after cut-elimination. This problem is very similar to the ones discussed in the Introduction. It is not at all clear how this renaming can be done for example after c-b-reduction (copying a box).

Another way out would be to extend the idea used in [5], similarly to deduction graphs: Change the $\forall$-rule and work with two kinds of variables. This might very well work.

Whence proof nets with quantifiers are defined properly and completely, it seems likely that we can define a reduction-preserving translation from $\mathtt{DG}_{\mathtt{U}}$s to them.

## 6 Conclusions

We have shown that deduction graphs, a generalisation of both Gentzen-Prawitz natural deduction and Fitch-style flag-deduction, can be extended with universal quantification without having to alter or to extend the underlying structure, the so-called closed box directed graphs.

We have discussed the problems with renaming that arise when eliminating cuts in these graphs, and we have presented a solution to this.

We have studied the process of cut-elimination, which consists of repeat-elimination, unsharing (including renaming), incorporation, and safe cut-elimination. Furthermore, we have demonstrated that the process of cut-elimination is strongly normalising.

[4] Personal communication Lorenzo Tortora de Falco.

Finally, by pointing out similarities between deduction graphs and proof nets, we have indicated some future work in the latter area.

## 7 Acknowledgements

We thank the referees for their useful comments.

## References

[1] Gerhard Gentzen. Untersuchungen über das logische Schliessen. In M.E. Szabo, editor, *Collected Papers of Gerhard Gentzen*. North-Holland Publishing Company, 1969.

[2] Herman Geuvers and Iris Loeb. From deduction graphs to proof nets: Boxes and sharing in the graphical presentation of deductions. In Rastislav Královič and Pawel Urzyczyn, editors, *MFCS*, volume 4162 of *Lecture Notes in Computer Science*, pages 39–57. Springer, 2006.

[3] Herman Geuvers and Iris Loeb. Natural Deduction via Graphs: Formal Definition and Computation Rules. *Mathematical Structures in Computer Science (Special Issue on Theory and Applications of Term Graph Rewriting)*, Volume 17(03):485–526, 2007.

[4] Jean-Yves Girard. Linear logic. *Theor. Comput. Sci.*, 50:1–102, 1987.

[5] Jean-Yves Girard. Quantifiers in linear logic II. In Giovanna Corsi and Giovanni Sambin, editors, *Atti della Congresso: Nuovi Problemi della Logica e della Filosofia della Scienza, Vol. 2*, pages 79–89. Bologna: CLUEB, 1991.

[6] Iris Loeb. *Natural Deduction: Sharing by Presentation*. PhD thesis, Radboud Universiteit Nijmegen, 2007.

[7] Dag Prawitz. *Natural Deduction: a proof-theoretical study*. Stockholm: Almqvist och Wiksell, 1965.

[8] Richmond H. Thomason. *Symbolic Logic: an Introduction*. New York: Macmillan, 1970.

[9] Lorenzo Tortora de Falco. *Réseaux, cohérence et expériences obsessionelles*. PhD thesis, Université Paris VII- Denis-Diderot, 2000.